Error Analysis

July 16, 2020

1 Overview

The error calculations are partially done in sfit4 and completed in the supplemental python code. Sfit4 calculates the Jacobian matrix ($\mathbf{K}$) and the Gain matrix ($\mathbf{G}$). It also assembles the a priori covariance matrix ($\boldsymbol{S_a}$). The Python code constructs the instrument covariance matrix ($\boldsymbol{S_e}$), calculates the averaging kernel ($\mathbf{A}$) and calculates the systematic, random, and total errors in various units. The errors are calculated as described in Rodgers. (2000); Rodgers and Connor. (2003).

2 Error Calculation

The retrieval $\hat{\boldsymbol{x}}$ is found by applying some sort of retrieval method to the measurements. This can be described as:

$$\hat{\boldsymbol{x}} = \boldsymbol{R}[\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{b}) + \boldsymbol{\Delta f}(\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{b'}) + \boldsymbol{\epsilon}, \hat{\boldsymbol{b}}, \boldsymbol{x_a}] \tag{2.1}$$

Where, $\boldsymbol{R}$ is the retrieval method, $\boldsymbol{F}$ is the forward model which approximates some forward function $\boldsymbol{f}$. The forward function describes the complete physics of the measurement and maps the actual state conditions to the measurements within some experimental error, $\boldsymbol{\epsilon}$.The state vector is given as $\boldsymbol{x}$ and $\boldsymbol{b}$ consists of other parameters not included in the state vector. The difference between the forward model $\boldsymbol{F}$ and the 'real' physics is given as $\boldsymbol{\Delta f}$ where $\boldsymbol{b'}$ are the parameters ignored in constructing the forward model. The best estimate of the forward function parameters is given as $\hat{\boldsymbol{b}}$, while $\boldsymbol{x_a}$ are parameters that do not appear in the forward function but still have an influence on the retrieval. The vector $\boldsymbol{x_a}$ is an a priori estimate of $\boldsymbol{x}$.

In order to understand the sources of error we linearize (2.1) in two steps. First we linearize the forward model about $\boldsymbol{x} = \boldsymbol{x_a}$ and $\boldsymbol{b} = \hat{\boldsymbol{b}}$ and then we linearize the actual inverse method $\boldsymbol{R}$ about $\boldsymbol{y} = \boldsymbol{y_a}$ using the first order term of a Taylor series expansion. Linearization of the forward model yields:

$$\begin{aligned} \hat{\boldsymbol{x}} =& \boldsymbol{R}[\boldsymbol{F}(\boldsymbol{x_a}, \hat{\boldsymbol{b}}) + \boldsymbol{K_x}(\boldsymbol{x} - \boldsymbol{x_a}) + \boldsymbol{K_b}(\boldsymbol{b} - \hat{\boldsymbol{b}}) + \\ & \boldsymbol{\Delta f}(\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{b'}) + \boldsymbol{\epsilon}, \hat{\boldsymbol{b}}, \boldsymbol{x_a}] \end{aligned} \tag{2.2}$$

Where, $\boldsymbol{K_x}$ and $\boldsymbol{K_b}$ are the sensitivities of the forward model to the state $(\boldsymbol{\partial F/\partial x})$ and forward model parameters $(\boldsymbol{\partial F/\partial b})$ respectively. Using equation (2.2), we linearize the retrieval or inverse method about the point $\boldsymbol{y_a}$ or $\boldsymbol{F(x_a, \hat{b})}$ to yield:

$$\begin{aligned}\hat{\boldsymbol{x}} =& \boldsymbol{R}[\boldsymbol{F}(\boldsymbol{x_a}, \hat{\boldsymbol{b}}), \hat{\boldsymbol{b}}, \boldsymbol{x_a}] + \boldsymbol{G_y}[\boldsymbol{K_x}(\boldsymbol{x} - \boldsymbol{x_a}) + \\ & \boldsymbol{K_b}(\boldsymbol{b} - \hat{\boldsymbol{b}}) + \boldsymbol{\Delta f}(\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{b'}) + \boldsymbol{\epsilon}]\end{aligned} \tag{2.3}$$

Where, $\boldsymbol{G_y}$ is the sensitivity of the retrieval to the measurement $(\boldsymbol{\partial R/\partial y})$. Assuming that the retrieval does not introduce a bias or $\boldsymbol{R}[\boldsymbol{F}(\boldsymbol{x_a}, \hat{\boldsymbol{b}}), \hat{\boldsymbol{b}}, \boldsymbol{x_a}] = \boldsymbol{x_a}$ and grouping the total error in the measured signal relative to the forward model such that $\boldsymbol{\epsilon_y} = \boldsymbol{K_b}(\boldsymbol{b} - \hat{\boldsymbol{b}}) + \boldsymbol{\Delta f}(\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{b'}) + \boldsymbol{\epsilon}$ we can rewrite (2.2) as:

$$\hat{\boldsymbol{x}} - \boldsymbol{x} = (\boldsymbol{A} - \boldsymbol{I_n})(\boldsymbol{x} - \boldsymbol{x_a}) \tag{2.4a}$$

$$+ \boldsymbol{G_y K_b}(\boldsymbol{b} - \hat{\boldsymbol{b}}) \tag{2.4b}$$

$$+ \boldsymbol{G_y \Delta f}(\boldsymbol{x}, \boldsymbol{b}, \boldsymbol{b'}) \tag{2.4c}$$

$$+ \boldsymbol{G_y \epsilon} \tag{2.4d}$$

Where $\boldsymbol{A}$ is the sensitivity of the retrieval to the true state, $\boldsymbol{A} = \boldsymbol{G_y K_x}$. The terms (2.4a), (2.4b), (2.4c), (2.4d) are the smoothing error, model parameter error, forward model error, and retrieval noise respectively.

2.1 Smoothing Error

The retrieval is actually an estimate of a state smoothed by the averaging kernel. The difference between these two states is given by the smoothing error. This type of error is considered a systematic error. In order to calculate the actual smoothing error, (2.4a), one must know the true state. In our error analysis we are looking for a statistical description of the error, which can be calculated from an ensemble of states. Taking the expectation value of (2.4a) we obtain the covariance matrix for the smoothing uncertainty:

$$\boldsymbol{S_s} = (\boldsymbol{A} - \boldsymbol{I})\, \boldsymbol{S_a}\, (\boldsymbol{A} - \boldsymbol{I})^{\boldsymbol{T}} \tag{2.5}$$

Where $\boldsymbol{S_a}$ is the covariance matrix of the prior estimate of $\boldsymbol{x}$. The full $\boldsymbol{S_a}$ matrix in sfit4 is constructed from the inputs in the sfit4.ctl file in the sfit4 core code. The averaging kernel $\boldsymbol{A}$ is also calculated in the sfit4 core code. The actual calculation of the smoothing error covariance matrix is done in the sfit4 python code.

Note: How is $\boldsymbol{G_y}$ and $\boldsymbol{K_Y}$ calculated in sfit4??

2.2 Forward Model Error

In practice the forward model error is difficult to calculate because it involves knowing the errors associated with physical parameterizations of the forward

model. In sfit4, this term is generally considered small and not included in the error analysis.

2.3 Model Parameter Error

The model parameter error represent the errors in the forward model parameters such as temperature, solar zenith angle, and spectroscopic parameters. These errors can contain both systematic and random components. We obtain the covariance matrix for this error by taking the expectation value of (2.4b), which yields:

$$\boldsymbol{S_f = G_y K_b S_b K_b^T G_y^T} \tag{2.6}$$

Where, $\boldsymbol{S_b}$ is the error covariance matrix on the forward model parameters. The $\boldsymbol{S_b}$ values are defined in the Sb.ctl file as described in section 4.0.2. The $\boldsymbol{K_b}$ matrix is calculated in the sfit4 core code and is described in section 3.1.

2.4 Retrieval Noise

The retrieval noise is a measure of the noise of the instrument and is characterized through the signal to noise (SNR) value of the instrument for the spectral region of interest. The error covariance matrix is found by taking the expectation value of (2.4d):

$$\boldsymbol{S_m = G_y S_\epsilon G_y^T} \tag{2.7}$$

Where, $\boldsymbol{S_\epsilon}$ is the covariance of the measurement noise. The $\boldsymbol{S_\epsilon}$ matrix is constructed in the sfit python code as described in 4.0.2.

3 Sfit4 Error Implementation

3.1 Calculation of The Weighting Function, K

Through the previous error analysis discussion we see that it is important to understand the influence of the state vector, $\boldsymbol{x}$, and forward model parameters, $\boldsymbol{b}$ on the forward model. This is accomplished through the weighting function or the $\boldsymbol{K}$ matrix. In regards to the state vector $\boldsymbol{x}$, the weighting function is $\boldsymbol{K_x}$ or $\boldsymbol{\partial F/\partial x}$. For the forward model parameters, the weighting function is $\boldsymbol{K_b}$ or $\boldsymbol{\partial F/\partial b}$. In further discussion we will use the state vector $\boldsymbol{x}$; however, note that this discussion can also be applied to the forward model parameters. The forward model of sfit4 consists of a radiative transfer model. Therefore, we can write the $\boldsymbol{K_x}$ matrix for each atmospheric layer, $\boldsymbol{k}$ as:

$$\boldsymbol{K_x(k) = \frac{\partial f(z_0)}{\partial b_k}} \tag{3.1}$$

Where, $\boldsymbol{f(z_0)}$ is the flux density or irradiance as seen by the instrument. Note that this $\boldsymbol{f}$ is different than the forward function. Only in this section will

$\boldsymbol{f}$ refer to the irradiance. Typically, in order to calculate the $\boldsymbol{K_x}$ matrix one would perturb each element of the state vector for each atmospheric layer and calculate a new spectral flux density.

4 Implementation of Error Calculation in sfit4 (> v1.07)

If running error analysis through Layer 1, the errFlg flag needs to be chosen inside the input layer 1 file. Additionally, the Kb needs to be True in the sfit4 control file. Below are some notes.

1.- The traditional single sb.ctl used in sfit4 v9.4.4 is not implemented in the latest sfit4 v1.0. Instead, a single sb control file is used for all gases.

2.- For a harmonized IRWG error calculation, in particular spectroscopy uncertainties, there is a default control file that the sfit4 development team has been created. We suggets to use this file.

3. This file can be found in the Layer1 folder (called sbDefaults.ctl).

4. To run error calculation the path to this file needs to be defined in the sfit4.ctl file as "file.in.sbdflt".

The sbDefaults.ctl file has several functions including controlling the outputs of the error analysis including units, specifying the inputs to the supplemental error analysis, and defining the diagonals of the $\boldsymbol{S_b}$ values.

Note: For sfit4 v1.0 the default configuration is to use the above approach. However, one might use the v9.4.4 approach by using the flag "-o" in Layer1 or Layer 0., e.g., to run error analysis through Layer 0: sfit4Layer0.py -fe -o.

4.0.1 Sb.ctl Output Parameters

Through the output flags in the Sb.ctl files one can specify the units of the output covariance matrices. Output matrices can be specified in units of *VMR* or *molecules* cm^{-2}. Output flags also control which covariance matrices are written, such as total, random, and systematic. In addition, parameters in the Sb.ctl file set the file names for the output files. If multiple sites are used and parameters are characterized for each site, e.g., temperature, one might need a single sb.ctl file for each gas.

4.0.2 Sb.ctl Input Parameters

The supplemental error calculation takes in inputs from the sfit4 core code. Currently the only optional input source for the supplemental calculations is

for the instrument covariance matrix $\mathbf{S_e}$. The diagonals of the $\mathbf{S_e}$ matrix are constructed using $\frac{\mathbf{1}}{\boldsymbol{SNR^2}}$. The off diagonal elements of the $\mathbf{S_e}$ matrix are not considered. From the Sb.ctl file there are two options for constructing the $\mathbf{S_e}$ matrix. If in the Sb.ctl file the SeInputFlg is set to true the supplemental calculations will use the sfit4 output specified in the file.out.seinv_vector field of the sfit4.ctl file. These values are the SNR used in the retrieval accounting for any weighting done in the sfit4.clt file. If the SeInputFlg is set to false the $\mathbf{S_e}$ matrix is constructed from the SNR values given in the sfit4 output summary files. These SNR values come directly from the t15asc files and are not influenced by the weighting of the SNR in the sfit4.ctl file.

The units for the Sb values are either native or fractional. A legend at the top of the Sb.ctl file indicates how to specify each parameter. There are several parameters in which the user can specify either native or fractional.

A handy presentation from B. Langerock with more details is found here: ftp://nitrogen.acom.ucar.edu//user/jamesw/sfit4/2019Workshop